Document Title	# Specification of Development Error Tracer
Document Owner	AUTOSAR GbR
Document Responsibility	AUTOSAR GbR
Document Identification No	017
Document Classification	Standard

Document Version	2.2.0
Document Status	Final
Part of Release	3.0
Revision	0001

Document Change History

Date	Version	Changed by	Change Description
12.12.2007	2.2.0	AUTOSAR Administration	• Added API GetVersionInfo to harmonize SWS with AUTOSAR conventions • Document meta information extended • Small layout adaptations made
25.01.2007	2.1.0	AUTOSAR Administration	• Added BSW00436 to traceability matrix • Added Memmap.h • Added Chapter 11 • Legal disclaimer revised • “Advice for users” revised • “Revision Information” added
17.05.2006	2.0.0	AUTOSAR Administration	Changed to new SWS template
08.07.2005	1.0.0	AUTOSAR Administration	Initial Release

Page left intentionally blank

Table of Contents

1 Introduction and functional overview

This specification describes the API of the Development Error Tracer. All detected development errors in the Basic Software are reported to this module. The API parameters allow for tracing source and kind of error:

- Module in which error has been detected
- Function in which error has been detected
- Type of error

The functionality behind the API of this module is not in scope of this specification. It is up to the software developer and software integrator to choose the optimal strategy for his specific application and testing environment. Possible functionalities could be:

- Set debugger breakpoint within error reporting API
- Count reported errors
- Log calls and passed parameters in RAM buffer
- Send reported errors via communication interface to external logger

2 Acronyms and abbreviations

Not applicable.

3 Related documentation

3.1 Input documents

[1] List of Basic Software Modules,
https://svn2.autosar.org/repos2/22_Releases/
AUTOSAR_BasicSoftwareModules.pdf

[2] Layered Software Architecture,
https://svn2.autosar.org/repos2/22_Releases/
AUTOSAR_LayeredSoftwareArchitecture.pdf

[3] General Requirements on Basic Software Modules
https://svn2.autosar.org/repos2/22_Releases/
AUTOSAR_SRS_General.pdf

[4] AUTOSAR Basic Software Module Description Template,
https://svn2.autosar.org/repos2/22_Releases/
AUTOSAR_BSW_Module_Description.pdf

3.2 Related standards and norms

Not applicable.

4 Constraints and assumptions

4.1 Limitations

This specification does not define the functionality behind the error reporting API.

Memory protection mechanisms of the operating system are not taken into account. It is assumed that the whole Basic Software runs in supervisor mode or the switch to supervisor mode is done by a system call within the error reporting function of the DET module.

4.2 Applicability to car domains

No restrictions.

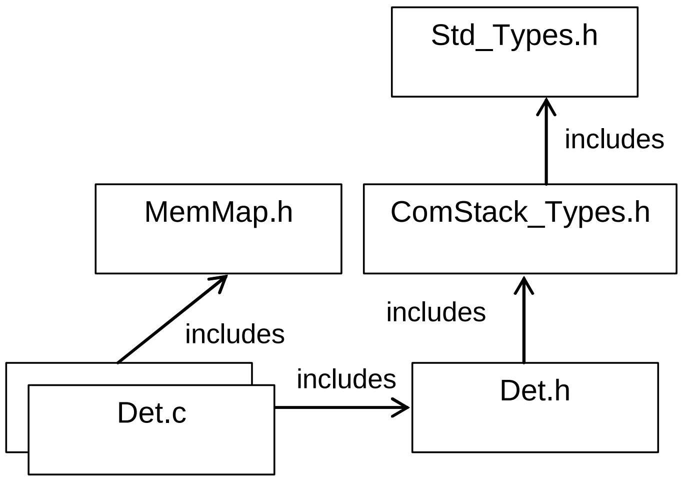
5 Dependencies to other modules

5.1 File structure

DET004: The DET module's source code shall offer a header file `Det.h` which includes all user relevant information for the Development Error Traces.

As the DET module does not define any module specific types, there is no requirement for additional include files.

DET006: The DET module's source code shall include the file `MemMap.h` in case the mapping of code and data to specific memory sections via memory mapping file is needed.



6 Requirements traceability

Only selected requirements related to the DET module are shown here. The remaining BSW General requirements have no relevance to this type of BSW module.

Requirement	***Satisfied by***
[BSW00300] Module naming convention [approved]	DET004
[BSW00301] Limit imported information [approved]	Not applicable (this is a development support module, no included files)
[BSW00302] Limit exported information [approved]	DET004
[BSW00304] AUTOSAR integer data types [approved]	Not applicable (Not used in this module)
[BSW00305] Self-defined data types naming convention [approved]	Not applicable (Not used in this module)
[BSW00306] Avoid direct use of compiler and platform specific keywords [approved]	Not applicable (Not used in this module)
[BSW00307] Global variables naming convention [approved]	Not applicable (Not used in this module)
[BSW00308] Definition of global data [approved]	Not applicable (Not used in this module)
[BSW00309] Global data with read-only constraint [approved]	Not applicable (Not used in this module)
[BSW00310] API naming convention [approved]	DET000 DET001 DET002
[BSW00312] Shared code shall be reentrant [approved]	Not applicable (Not used in this module)
[BSW00314] Separation of interrupt frames and service routines [approved]	Not applicable (Not used in this module)
[BSW00318] Format of module version numbers [approved]	DET003
[BSW00321] Enumeration of module version numbers [approved]	DET004
[BSW00324] Do not use HIS I/O Library [approved]	Not applicable (Not used in this module)
[BSW00325] Runtime of interrupt service routines [approved]	Not applicable (Not used in this module)
[BSW00326] Transition from ISRs to OS tasks [approved]	Not applicable (Not used in this module)
[BSW00327] Error values naming convention [approved]	Not applicable (Not used in this module)
[BSW00328] Avoid duplication of code [approved]	Not applicable (Not used in this module)
[BSW00329] Avoidance of generic interfaces [approved]	Not applicable (Not used in this module)
[BSW00330] Usage of macros / inline functions instead of functions [approved]	Not applicable (Not used in this module)
[BSW00331] Separation of error and status values [approved]	Not applicable (Not used in this module)
[BSW00333] Documentation of callback function context [approved]	Not applicable (Not used in this module)

[BSW00334] Provision of XML file [approved]	Not applicable (Not used in this module)
[BSW00335] Status values naming convention [approved]	Not applicable (Not used in this module)
[BSW00341] Microcontroller compatibility documentation [approved]	Not applicable (Not used in this module)
[BSW00342] Usage of source code and object code [approved]	Not applicable (Not used in this module)
[BSW00343] Specification and configuration of time [approved]	Not applicable (Not used in this module)
[BSW00341] Microcontroller compatibility documentation [approved]	Not applicable (Not used in this module)
[BSW00346] Basic set of module files [approved]	Not applicable (Not used in this module)
[BSW00347] Naming separation of different instances of BSW drivers [approved]	Not applicable (Not used in this module)
[BSW00350] Development error detection keyword [approved]	Not applicable (Not used in this module)
[BSW00353] Platform specific type header [approved]	Not applicable (Not used in this module)
[BSW00355] Do not redefine AUTOSAR integer data types [approved]	Not applicable (Not used in this module)
[BSW00350] Development error detection keyword [approved]	Not applicable (Not used in this module)
[BSW00358] Return type of init() functions [approved]	DET000
[BSW00359] Return type of callback functions [approved]	Not applicable (Not used in this module)
[BSW00360] Parameters of callback functions [approved]	Not applicable (Not used in this module)
[BSW00361] Compiler specific language extension header [approved]	Not applicable (Not used in this module)
[BSW00370] Separation of callback interface from API [approved]	Not applicable (Not used in this module)
[BSW00371] Do not pass function pointers via API [approved]	Not applicable (Not used in this module)
[BSW00373] Main processing function naming convention [approved]	Not applicable (Not used in this module)
[BSW00374] Module vendor identification [approved]	DET003
[BSW00376] Return type and parameters of main processing functions [approved]	Not applicable (Not used in this module)
[BSW00377] Module specific API return types [approved]	Not applicable (Not used in this module)
[BSW00378] AUTOSAR boolean type [approved]	Not applicable (Not used in this module)
[BSW00379] Module identification [approved]	Not applicable (Not used in this module)
[BSW00401] Documentation of multiple instances of configuration parameters [approved]	Not applicable (Not used in this module)
[BSW00408] Configuration parameter naming convention [approved]	Not applicable (Not used in this module)
[BSW00410] Compiler switches shall have defined values [approved]	Not applicable (Not used in this module)
[BSW00411] Get version info keyword [approved]	Not applicable (Not used in this module)
[BSW00413] Accessing instances of BSW modules [approved]	Not applicable (Not used in this module)
[BSW00414] Parameter of init function [approved]	DET000
[BSW00415] User dependent include files [approved]	Not applicable (Not used in this module)

[BSW005] No hard coded horizontal interfaces within MCAL [approved]	Not applicable (Not used in this module)
[BSW006] Platform independency [approved]	Not applicable (Not used in this module)
[BSW007] HIS MISRA C [approved]	Not applicable (Not used in this module)
[BSW009] Module User Documentation [approved]	Not applicable (Not used in this module)
[BSW010] Memory resource documentation [approved]	Not applicable (Not used in this module)
[BSW158] Separation of configuration from implementation [approved]	Not applicable (Not used in this module)
[BSW160] Human-readable configuration data [approved]	Not applicable (Not used in this module)
[BSW161] Microcontroller abstraction [approved]	Not applicable (Not used in this module)
[BSW162] ECU layout abstraction [approved]	Not applicable (Not used in this module)
[BSW164] Implementation of interrupt service routines [approved]	Not applicable (Not used in this module)
[BSW172] Compatibility and documentation of scheduling strategy [approved]	Not applicable (Not used in this module)
[BSW00344] Reference to link-time configuration	Not applicable (Not used in this module)
[BSW00404] Reference to post build time configuration	Not applicable (Not used in this module)
[BSW00405] Reference to multiple configuration sets	Not applicable (Not used in this module)
[BSW00345] Pre-compile-time configuration	Not applicable (Not used in this module)
[BSW159] Tool-based configuration	Not applicable (this is a tool requirement)
[BSW167] Static configuration checking	Not applicable (Requirement on configuration tool)
[BSW171] Configurability of optional functionality	Not applicable (Not used in this module)
[BSW170] Data for reconfiguration of AUTOSAR SW-Components	Not applicable (Not used in this module)
[BSW00380] Separate C-Files for configuration parameters	Not applicable (Not used in this module)
[BSW00419] Separate C-Files for pre-compile time configuration parameters	Not applicable (Not used in this module)
[BSW00381] Separate configuration header file for pre-compile time parameters	Not applicable (Not used in this module)
[BSW00412] Separate H-File for configuration parameters	Not applicable (Not used in this module)
[BSW00383] List dependencies of configuration files	Not applicable (Not used in this module)
[BSW00384] List dependencies to other modules	Not applicable (Not used in this module)
[BSW00387] Specify the configuration class of callback function	Not applicable (Not used in this module)
[BSW00388] Introduce containers	Not applicable (Not used in this module)
[BSW00389] Containers shall have names	Not applicable (Not used in this module)

[BSW00390] Parameter content shall be unique within the module	Not applicable (Not used in this module)
[BSW00391] Parameter shall have unique names	Not applicable (Not used in this module)
[BSW00392] Parameters shall have a type	Not applicable (Not used in this module)
[BSW00393] Parameters shall have a range	Not applicable (Not used in this module)
[BSW00394] Specify the scope of the parameters	Not applicable (Not used in this module)
[BSW00395] List the required parameters (per parameter)	Not applicable (Not used in this module)
[BSW00396] Configuration classes	Not applicable (Not used in this module)
[BSW00397] Pre-compile-time parameters	Not applicable (Not used in this module)
[BSW00398] Link-time parameters	Not applicable (Not used in this module)
[BSW00399] Loadable Post-build time parameters	Not applicable (Not used in this module)
[BSW00400] Selectable Post-build time parameters	Not applicable (Not used in this module)
[BSW00402] Published information	Partly fulfilled by DET003. Vendor version number for this header file not necessary.
[BSW00375] Notification of wake-up reason	Not applicable (Not used in this module)
[BSW101] Initialization interface	DET000 DET001 DET002
[BSW003] Version identification	DET003
[BSW004] Version check	DET003
[BSW00416] Sequence of Initialization	Not applicable (Not used in this module)
[BSW00406] Check module initialization	Not applicable (Not used in this module)
[BSW168] Diagnostic Interface of SW components	Not applicable (Not used in this module)
[BSW00407] Function to read out published parameters	Not applicable (Not used in this module)
[BSW00423] Usage of SW-C template to describe BSW modules with AUTOSAR Interfaces	Not applicable (Not used in this module)
[BSW00424] BSW main processing function task allocation	Not applicable (Not used in this module)
[BSW00425] Trigger conditions for schedulable objects	Not applicable (Not used in this module)
[BSW00426] Exclusive areas in BSW modules	Not applicable (Not used in this module)
[BSW00427] ISR description for BSW modules	Not applicable (Not used in this module)
[BSW00428] Execution order dependencies of main processing functions	Not applicable (not related to this specification)
[BSW00429] Restricted BSW OS functionality access	Not applicable (Not used in this module)
[BSW00431] The BSW Scheduler module implements task bodies	Not applicable (Not used in this module)
[BSW00432] Modules should have separate main processing functions for read/receive and write/transmit data path	Not applicable (Not used in this module)

[BSW00433] Calling of main processing functions	Not applicable (Not used in this module)
[BSW00434] The Schedule Module shall provide an API for exclusive areas	Not applicable (Not used in this module)
[BSW00336] Shutdown interface	Not applicable (Not used in this module)
[BSW00337] Classification of errors	Not applicable (this module does not produce own errors)
[BSW00338] Detection and Reporting of development errors	Not applicable (this module does not produce own errors)
[BSW00369] Do not return development error codes via API	Not applicable (Not used in this module)
[BSW00339] Reporting of production relevant error status	Not applicable (Not used in this module)
[BSW00348] Standard type header	Not applicable (Not used in this module)
[BSW00357] Standard API return type	Not applicable (Not used in this module)
[BSW00421] Reporting of production relevant error events	Not applicable (Not used in this module)
[BSW00422] Debouncing of production relevant error status	Not applicable (not related to this specification)
[BSW00420] Production relevant error event rate detection	Not applicable (not related to this specification)
[BSW00417] Reporting of Error Events by Non-Basic Software	Not applicable (Not used in this module)
[BSW00323] API parameter checking	Not applicable (Not used in this module)
[BSW004] Version check	Not applicable (Not used in this module)
[BSW00409] Header files for production code error IDs	Not applicable (Not used in this module)
[BSW00385] List possible error notifications	Not applicable (Not used in this module)
[BSW00386] Configuration for detecting an error	Not applicable (Not used in this module)
[BSW00436] Module Header File Structure for the Basic Software Memory Mapping	DET006

7 Functional specification

Not in scope of this specification.

8 API specification

8.1 Type definitions

No module specific types are required.

8.2 Function definitions

8.2.1 Det_Init

DET008:

Service name:	Det_Init

Syntax:	`void Det_Init(`
	
	 `)`

Service ID[hex]:	0x00
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	Service to initialize the Development Error Tracer.

DET000: The initialization function `Det_Init` shall reset all error counters and error logging data.

After invocation of the function `Det_Init`, the DET has been partly initialized but the complete functionality is not present (e.g. due to missing RAM mirrors).

Note: If an initialization of error information (e.g. after reset) is not desired, the environment from the DET module shall not call the function Det_Init.

8.2.2 Det_ReportError

DET009:

Service name:	Det_ReportError

Syntax:	`void Det_ReportError(`
	 `    uint16 ModuleId,`
	 `    uint8 InstanceId,`
	 `    uint8 ApiId,`
	 `    uint8 ErrorId`
	 `)`

Service ID[hex]:	0x01
Sync/Async:	Depending on implemented functionality: 1. Breakpoint set: no return 2. Internal error counting/logging in RAM: synchronous 3. External error logging via communication interface: asynchronous

Reentrancy:	Reentrant	
Parameters (in):	ModuleId	Module ID of calling module.
	InstanceId	The identifier of the index based instance of a module, starting from 0, If the module is a single instance module it shall pass 0 as the InstanceId.
	ApiId	ID of API service in which error is detected (defined in SWS of calling module)
	ErrorId	ID of detected development error (defined in SWS of calling module).
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Service to report development errors.	

DET001: The DET module's environment shall use the function `Det_ReportError` to report development errors.

Note: The way that the error is handled varies, both in terms of implementation (breakpoint, RAM storage, etc.) and the initialization stage. For example, if the error tracer sends the events via CAN, they can be stored in RAM until `Det_Start` is called.

8.2.3 Det_Start

DET010:

Service name:	Det_Start

Syntax:	`void Det_Start(`
	
	`)`

Service ID[hex]:	0x02
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	Service to start the Development Error Tracer.

DET002: The DET module's environment shall use the function `Det_Start` to start the DET module after it has been initialized by `Det_Init`.

Note: In case the DET does not require separated init calls, e.g. no RAM mirrors are required, the `Det_Start` function can be empty. This means that both `Det_Init` and `Det_Start` shall always be available.

8.2.4 Det_GetVersionInfo

DET011:

Service name:	Det_GetVersionInfo	

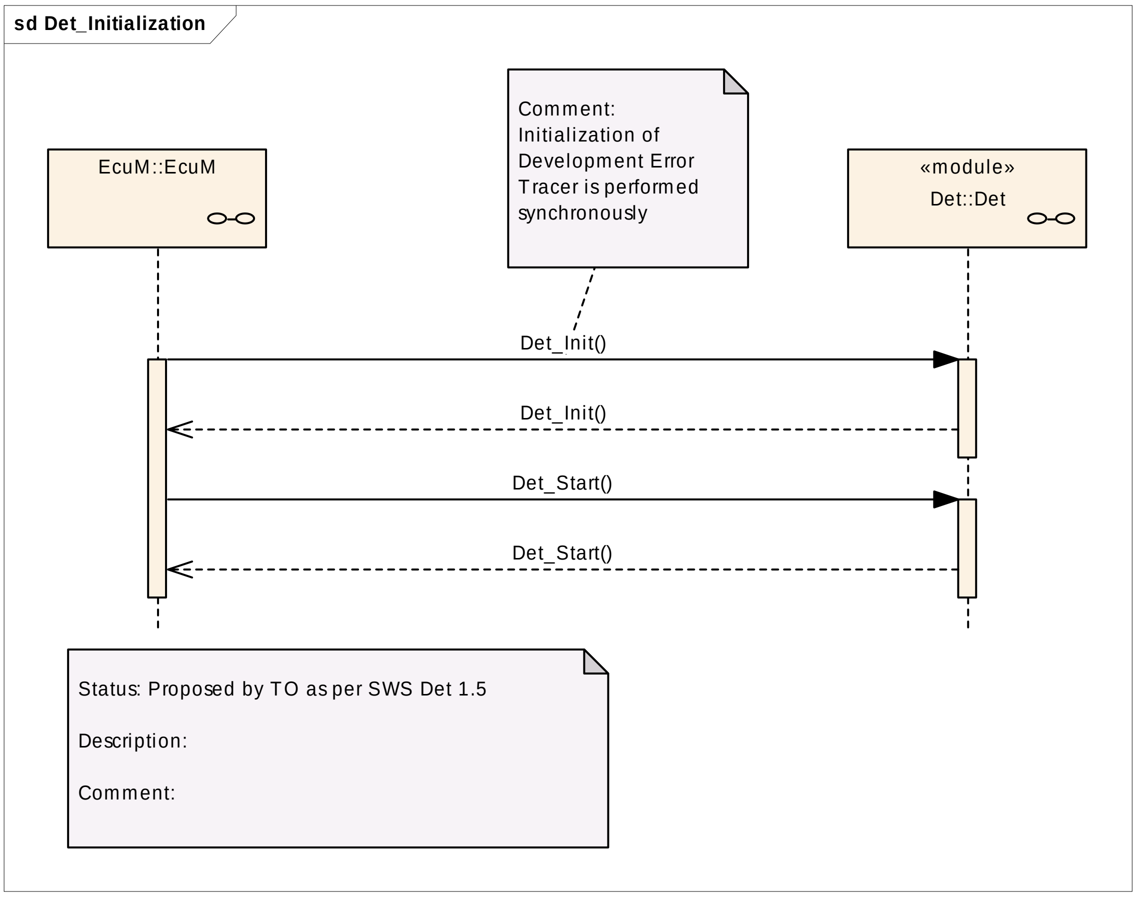
Syntax:	`void Det_GetVersionInfo(`
	`    Std_VersionInfoType* versioninfo`
	`)`	

Service ID[hex]:	0x03	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	versioninfo	Pointer to where to store the version information of this module.
Return value:	None	
Description:	Returns the version information of this module.	

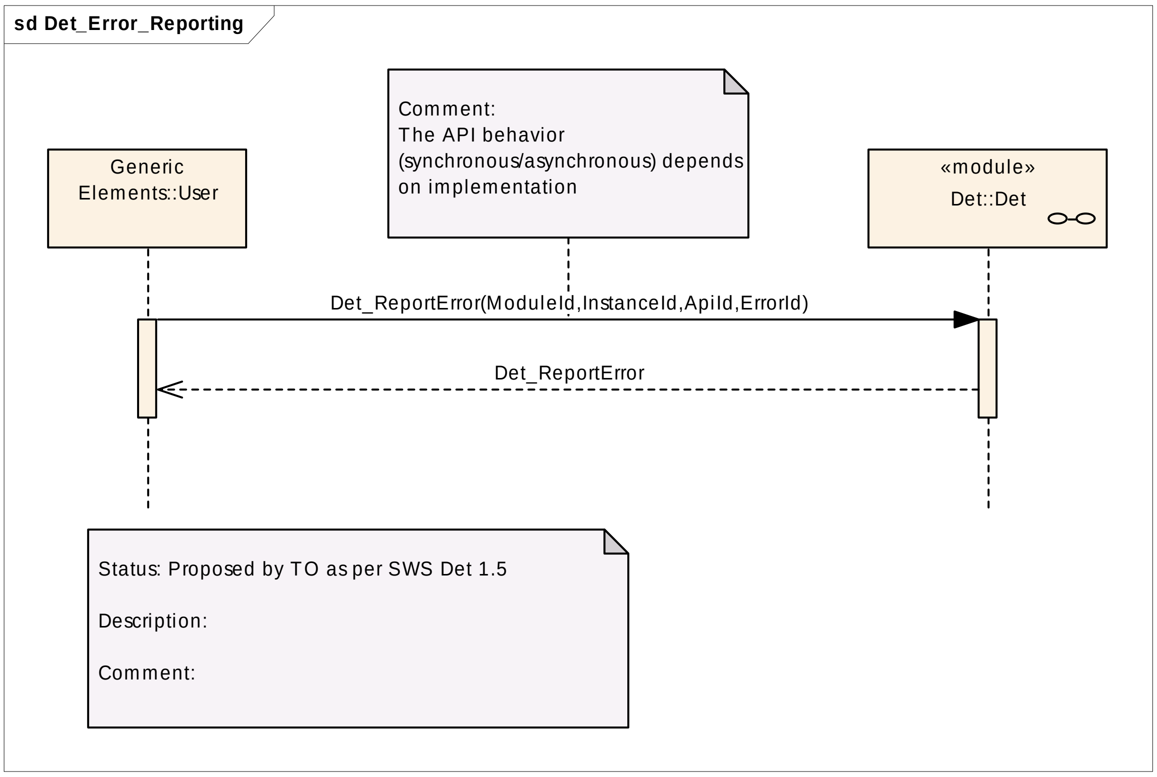
DET012: The function `Det_GetVersionInfo` shall return the version information of this module. The version information includes:
- Module Id
- Vendor Id
- Vendor specific version number.

9 Sequence diagrams

9.1 Initialization



9.2 Error reporting



10 Configuration specification

10.1 Containers and configuration parameters

10.1.1 Det

Module Name	**Det**
Module Description	Configuration of the Det (Development Error Tracer) module. There are NO standardized configuration parameters defined. All the configuration of the Det is done via vendor-specific configuration parameters which need to be defined inside this ModuleDef.

No Included Containers

10.2 Published Information

Published information contains data defined by the implementer of the SW module that does not change when the module is adapted (i.e. configured) to the actual HW/SW environment. It thus contains version and manufacturer information.

The standard common published information like

vendorId (<Module>_VENDOR_ID),
moduleId (<Module>_MODULE_ID),
arMajorVersion (<Module>_AR_MAJOR_VERSION),
arMinorVersion (<Module>_ AR_MINOR_VERSION),
arPatchVersion (<Module>_ AR_PATCH_VERSION),
swMajorVersion (<Module>_SW_MAJOR_VERSION),
swMinorVersion (<Module>_ SW_MINOR_VERSION),
swPatchVersion (<Module>_ SW_PATCH_VERSION),
vendorApiInfix (<Module>_VENDOR_API_INFIX)

is provided in the BSW Module Description Template (see [4] Figure 4.1 and Figure 7.1).

Additional published parameters are listed below if applicable for this module.

11 AUTOSAR Interfaces

11.1 Scope of this Chapter

This chapter defines the AUTOSAR Interfaces of the Development Error Tracer Service (DET).

The definitions in this section are interpreted to be in `ARPackage AUTOSAR/Services/DET`.

11.2 Overview

The DET BSW module was originally developed for supporting the other BSW modules during development phase. It is however possible to also use the DET to perform error tracing during development of SW-C's.

11.2.1 Use Case

On each ECU, there are one instance of the DET Service and several Atomic Software Component instances named “clients” which are interacting with the DET Service.

Each component instance may report a development error to the DET Service, and these errors can then be analyzed with debugging tools. The behavior of the DET is not standardized. Thus, what happens when using DET Service might differ on different ECUs or different DET implementations. The DET might for example do one of the following:

- Set debugger breakpoint within error reporting API
- Count reported errors
- Log calls and passed parameters in RAM buffer
- Send reported errors via communication interface to external logger

11.3 Specification of the Ports and Port Interfaces

This chapter specifies the ports and port interfaces which are needed in order to operate the DET functionality over the VFB.
Each AUTOSAR SW-C which uses the service must contain “service ports” in its own SW-C description which will be typed by the same interfaces and which has to be connected to the ports of the DET, so that the RTE, the appropriate IDs and the required symbols can be generated.

11.3.1 General Approach

The client-server paradigm is used since more than one parameter has to be transferred.

In order to reuse the C API already defined in the DET BSW module, the DET service uses the same argument names as in the C API, even though the names can not directly be mapped into the SW-C world. "Module ID" can preferably be interpreted as either a component or runnable entity but this is the decision of the implementer of the SW-C.

The DET service needs a "Module ID" as first argument for the C-function.
In order to keep the client code independent from the configuration of number of clients, the "Module IDs" are not passed from the clients to DET but are modeled as "port defined argument values" of the Provide ports on the DET side. As a consequence, the "Module IDs" will not show up as arguments in the operation of the client-server interface. As a further consequence for this approach, there will be separate ports for each "Module ID" both on the client side as well as on the server side.
The Module ID type is of range 0…65536. Values in the range of 0...255 are reserved for Basic Software Modules, all others can be used for application software components.

11.3.2 Data Types

For the port interface of the DET service only `Uint8` is required.

11.3.3 Port Interface

There is only one operation used as service from DET. In C-style, it looks as follows:

```
void Det_ReportError
(
   uint16 ModuleId,
   uint8  InstanceId,
   uint8  ApiId,
   uint8  ErrorId
)
```

That operation looks like the following in Port Interface pseudo code.

```
ClientServerInterface DETService {
      isService = true;
      ReportError (     IN Uint8 InstanceId,
                        IN Uint8 ApiId,
                        IN Uint8 ErrorId);
};
```

Compared to the API, the “`Det_`” prefix in the name is not required, because the name given here will show up in the XML not globally but as part of an interface description.

Note that the argument `InstanceId` can not be used for component instance since only the first argument (in this case `ModuleId`) can be used in "port defined

argument value". Instead, multiple instantiations of a module are converted into multiple "Module IDs".
It is up to the implementer of a SW-C to decide about the semantics of the arguments `InstanceId`, `ApiId` and `ErrorId` since these arguments are not standardized by AUTOSAR (at least not above the RTE). However, the `ApiId` typically corresponds to the operations that can report an error, and `ErrorId` corresponds to the type of error that is reported. Both `ApiId` and `ErrorId` are numbered `0x00..0xFF` without specific order.

11.3.4 Ports

Figure 1 shows how AUTOSAR Software components (single or multiple instances) are connected via service ports to the DET.

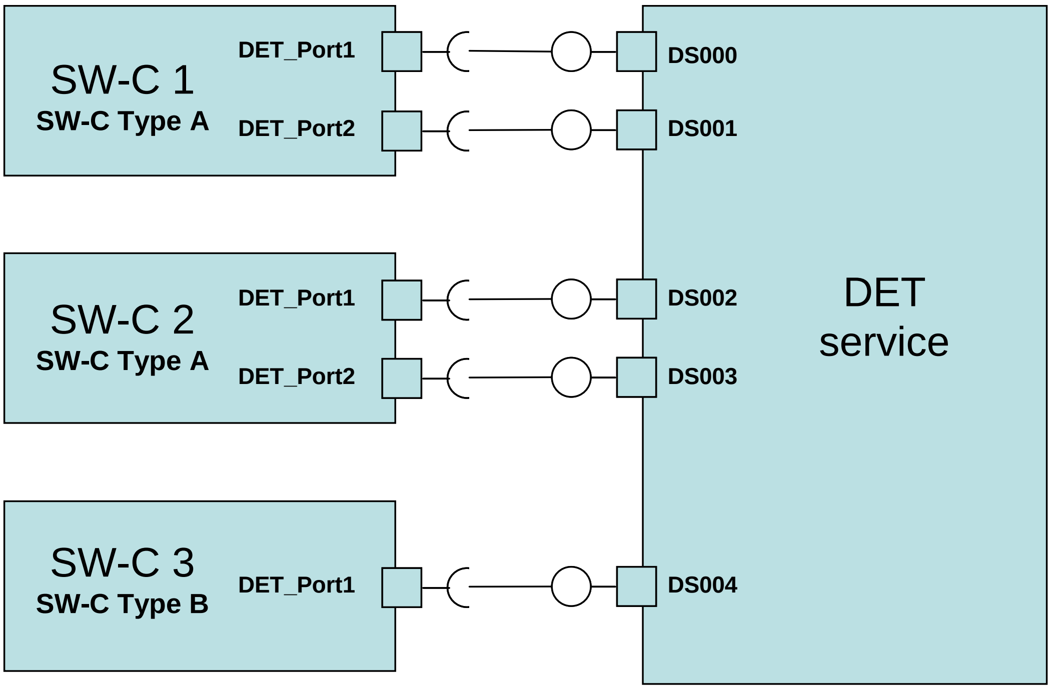


Figure 1: Example of SW-Cs connected to the DET via service ports. On the left side, there are two instances of component SW-C Type A and one instance of component SW-C Type B.

On the DET side, there is one port per "Module ID" providing the service of the interface `DETService` described above. Each client has one port for requiring the service for each "Module ID" associated with that Software Component (SWC or BSW-mudule).
Typically one port is created for each SW-C. It is also possible to create a port for each runnable entity inside a SW-C.

There is no naming convention for the ports providing the services. In this example they are named: `DS000, DS001, … , DSnnn`

The port values shall be converted into an `Uint16`, with values between `0x1000` and `0xFFFF`, in order to not interfere with BSW Module IDs.

11.4 Definition of the Service

The Provide Ports have a certain relation to the `InternalBehavior` of the DET: With each call, the "Module ID" is passed as an additional argument by the RTE to the C-function which implements the associated runnable entity (feature "port defined argument value").

```
Service DET
{
      // "nnn" below is number of ports using the service
      ProvidePort DETService DS000;
      ...
      ProvidePort DETService DSnnn;

      InternalBehavior
      {
            RunnableEntity ReportError
                  symbol "Det_ReportError"
                  canbeInvokedConcurrently = TRUE

            DS000.ReportError->ReportError
            PortArgument {port=DS000, value.type=Uint16,
            value.value=0x1000}
            …
            DSnnn.ReportError->ReportError
            PortArgument {port=DSnnn, value.type=Uint16,
            value.value=(0x1000 + nnn)}
      };
};
```

11.5 Configuration of the DET

The "Module IDs" of the DET service are modeled as "port defined argument values". Thus the configuration of those values is part of the RTE configuration. Pre-compile configuration can be done by changing the XML specification for the ports on the client (SW-C) or service (i.e. DET) side.